Machine Learning

Basics of ML

Computer & Health Informatics Department

What is Machine Learning?

- Machine Learning (ML) is that field of computer science with the help of which computer systems can provide sense to data in much the same way as human beings do.
- In simple words, ML is a type of artificial intelligence that extract patterns out of raw data by using an algorithm or method.
- The main focus of ML is to allow computer systems learn from experience without being explicitly programmed or human intervention.

Need for Machine Learning

- Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems.
- AI is still in its initial stage and haven't surpassed human intelligence in many aspects.
- what is the need to make machine learn?

- The most suitable reason for doing this is, "to make decisions, based on data, with efficiency and scale".

Need for Machine Learning

- Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems.
- We can call it data-driven decisions taken by machines, particularly to automate the process.
- These data-driven decisions can be used, instead of using programing logic, in the problems that cannot be programmed inherently.
- The fact is that we can't do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale.
- Why the need for machine learning arises.

Why & When to Make Machines Learn?

- what scenarios we must make the machine learn?

- There can be several circumstances where we need machines to take data-driven decisions with efficiency and at a huge scale.

- The followings are some of such circumstances where making machines learn would be more effective.

Why & When to Make Machines Learn?

- **Lack of human expertise:** The scenario is in a domain where there is a lack of human expertise

- **Dynamic scenarios:** the scenarios which keep changing over time. Some of the examples can be network connectivity and availability of infrastructure in an organization.

Why & When to Make Machines Learn?

- **Difficulty in translating expertise into computational tasks:**

There can be various domains in which humans have their expertise,; however, they are unable to translate this expertise into computational tasks. In such circumstances we want machine learning. The examples can be the domains of speech recognition, cognitive tasks etc.

Machine Learning Model

- **The following formal definition of ML**
- "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."
- The above definition is basically focusing on three parameters, also the main components of any learning algorithm, namely Task(T), Performance(P) and experience (E).

Machine Learning Model

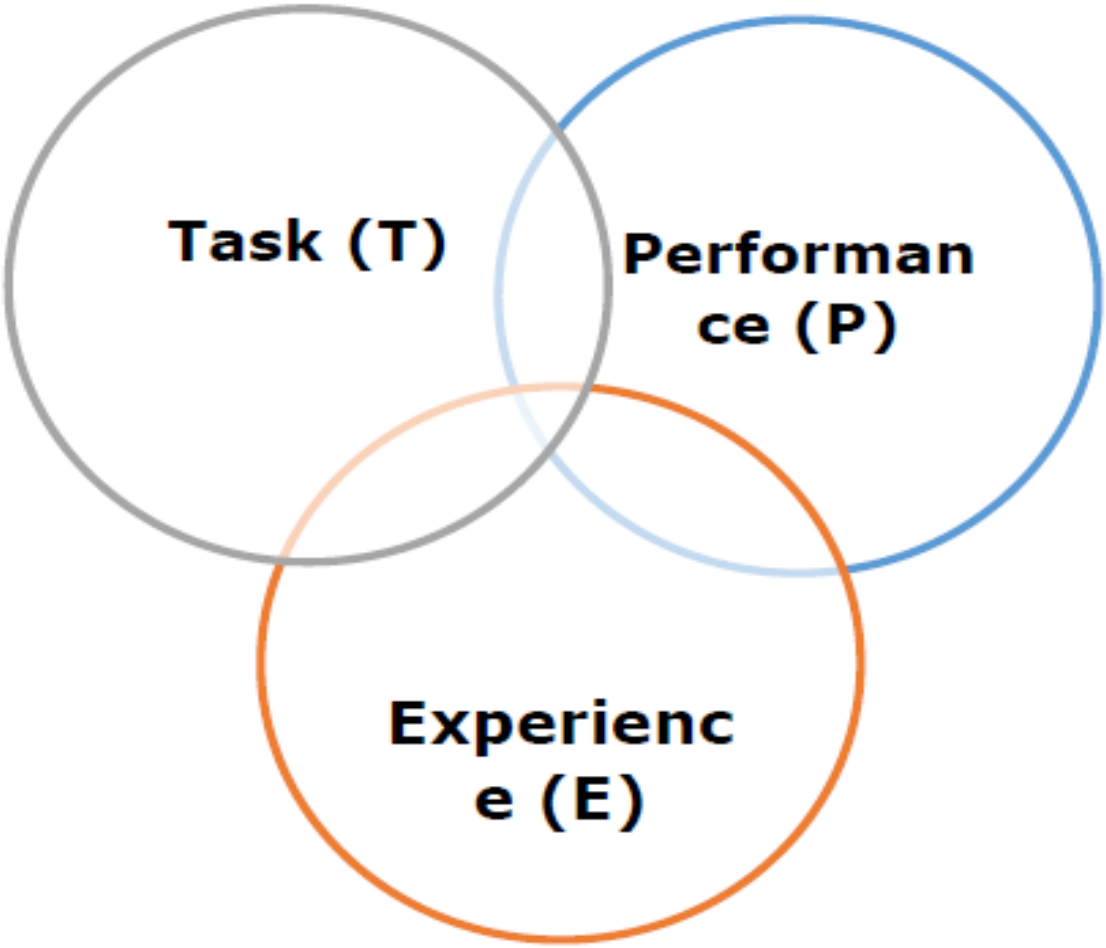
- In this context, we can simplify this definition as :

  ML is a field of AI consisting of learning algorithms that :

  - Improve their performance (P)
  - Executing some task (T)
  - Over time with experience (E)

- Based on the above, the following diagram represents a Machine Learning Model :

Machine Learning Model



Machine Learning Model

- Task(T)
  - The task T as the real-world problem to be solved. The problem can be anything like finding best house price in a specific location or to find best marketing strategy etc.
  - In machine learning, the definition of task is different because it is difficult to solve ML based tasks by conventional programming approach.
  - A task T is said to be a ML based task when it is based on the process and the system must follow for operating on data points.
  - The examples of ML based tasks are Classification, Regression, Structured annotation, Clustering, Transcription etc.

Machine Learning Model

- Experience (E)
  - As name suggests, it is the knowledge gained from data points provided to the algorithm or model.
  - Once provided with the dataset, the model will run iteratively and will learn some inherent pattern.
  - The learning thus acquired is called experience(E). Making an analogy with human learning, we can think of this situation as in which a human being is learning or gaining some experience from various attributes like situation, relationships etc.
  - Supervised, unsupervised and reinforcement learning are some ways to learn or gain experience.
  - The experience gained by out ML model or algorithm will be used to solve the task T.

Machine Learning Model

- Performance (P)
  - An ML algorithm is supposed to perform task and gain experience with the passage of time.
  - The measure which tells whether ML algorithm is performing as per expectation or not is its performance (P).
  - P is basically a quantitative metric that tells how a model is performing the task, T, using its experience, E. There are many metrics that help to understand the ML performance, such as accuracy score, F1 score, confusion matrix, precision, recall, sensitivity etc.

Challenges in Machines Learning

- While Machine Learning is rapidly evolving, making significant progresses with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are :
- **Quality of data** – Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.

Challenges in Machines Learning

- **Time-Consuming task** – Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.
- **Lack of specialist persons** – As ML technology is still in its beginning stage, availability of expert resources is a hard job.
- **No clear objective for formulating business problems** – Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology is not that mature yet.

Challenges in Machines Learning

- **Issue of overfitting & underfitting** – If the model is overfitting or underfitting, it cannot be represented well for the problem.
- **Curse of dimensionality** – Another challenge ML model faces is too many features of data points. This can be a real interruption.
- **Difficulty in deployment** – Complexity of the ML model makes it quite difficult to be deployed in real life.

Applications of Machines Learning

- Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML.
- It is used to solve many real-world complex problems which cannot be solved with traditional approach. Following are some real-world applications of ML :
  - Emotion analysis
  - Sentiment analysis
  - Error detection and prevention
  - Weather forecasting and prediction
  - Stock market analysis and forecasting

Applications of Machines Learning

- Speech synthesis
- Speech recognition
- Customer segmentation
- Object recognition
- Fraud detection
- Fraud prevention
- Recommendation of products to customer in online shopping

Python List

- List is an ordered sequence of items. It is one of the most used datatype in Python and is very flexible. All the items in a list do not need to be of the same type.
- Declaring a list is pretty straight forward. Items separated by commas are enclosed within brackets [ ].

```
>>> a = [1, 2.2, 'python']
```

- We can use the slicing operator [ ] to extract an item or a range of items from a list. Index starts form 0 in Python.

```
a = [5,10,15,20,25,30,35,40]
# a[2] = 15
print("a[2] = ", a[2])
# a[0:3] = [5, 10, 15]
print("a[0:3] = ", a[0:3])
# a[5:] = [30, 35, 40]
print("a[5:] = ", a[5:])
```

Python List

- Lists are mutable, meaning, value of elements of a list can be altered.

```
>>> a = [1,2,3]
>>> a[2]=4
>>> a
[1, 2, 4]
```

Python Tuple

- Tuple is an ordered sequence of items same as list . The only difference is that tuples are immutable. Tuples once created cannot be modified.

- Tuples are used to write-protect data and are usually faster than list as it cannot change dynamically.
- It is defined within parentheses () where items are separated by commas.

```
>>>  t = (5,'program', 1+3j)
```

- We can use the slicing operator [] to extract items but we cannot change its value.

Python Tuple

```
t = (5,'program', 1+3j)
# t[1] = 'program'
print("t[1] = ", t[1])
# t[0:3] = (5, 'program', (1+3j))
print("t[0:3] = ", t[0:3])
# Generates error
# Tuples are immutable
t[0] = 10
```

Python Strings

- String is sequence of Unicode characters. We can use single quotes or double quotes to represent strings.
- Multi-line strings can be denoted using triple quotes, ''' or """.

```
>>> s = "This is a string"
>>> s = '''a multiline
```

- Like list and tuple, slicing operator [ ] can be used with string. Strings are immutable.

```
s = 'Hello world!'
# s[4] = 'o'
print("s[4] = ", s[4])
# s[6:11] = 'world'
print("s[6:11] = ", s[6:11])
# Generates error
# Strings are immutable in Python
s[5] ='d'
```

Python Set

- Set is an unordered collection of unique items. Set is defined by values separated by comma inside braces { }. Items in a set are not ordered.

```
a = {5,2,3,1,4}
# printing set variable
print("a = ", a)
# data type of variable a
print(type(a))
```

- We can perform set operations like union, intersection on two sets. Set have unique values. They eliminate duplicates.

Python Set

```
>>> a = {1,2,2,3,3,3}
>>> a
{1, 2, 3}
```

- Since, set are unordered collection, indexing has no meaning. Hence the slicing operator [] does not work.

```
>>> a = {1,2,3}
>>> a[1]
Traceback (most recent call last):
  File "<string>", line 301, in runcode
  File "<interactive input>", line 1, in <module>
TypeError: 'set' object does not support indexing
```

Python Dictionary

- Dictionary is an unordered collection of key-value pairs.
- It is generally used when we have a huge amount of data. Dictionaries are optimized for retrieving data. We must know the key to retrieve the value.
- In Python, dictionaries are defined within braces {} with each item being a pair in the form key:value. Key and value can be of any type.

```
>>> d = {1:'value','key':2}
>>> type(d)
<class 'dict'>
```

- We use key to retrieve the respective value. But not the other way around

```
d = {1:'value','key':2}
print(type(d))
print("d[1] = ", d[1]);
print("d['key'] = ", d['key']);
# Generates error
print("d[2] = ", d[2]);
```

Conversion between data types

- We can convert between different data types by using different type conversion functions like int(), float(), str() etc.

```
>>> float(5)
5.0
```

- Conversion from float to int will truncate the value (make it closer to zero).

```
>>> int(10.6)
10
>>> int(-10.6)
-10
```

- Conversion to and from string must contain compatible values.

```
>>> float('2.5')
2.5
>>> str(25)
'25'
>>> int('1p')
Traceback (most recent call last):
  File "<string>", line 301, in runcode
  File "<interactive input>", line 1, in <module>
ValueError: invalid literal for int() with base 10: '1p'
```

Conversion between data types

- We can even convert one sequence to another.

```
>>> set([1,2,3])
{1, 2, 3}
>>> tuple({5,6,7})
(5, 6, 7)
>>> list('hello')
['h', 'e', 'l', 'l', 'o']
```

- To convert to dictionary, each element must be a pair

```
>>> dict([[1,2],[3,4]])
{1: 2, 3: 4}
>>> dict([(3,26),(4,44)])
{3: 26, 4: 44}
```

Examples: Python Program to Add Two Numbers

```
# This program adds two numbers
num1 = 1.5
num2 = 6.3
# Add two numbers
sum = float(num1) + float(num2)
# Display the sum
print('The sum of {0} and {1} is {2}'.format(num1, num2, sum))
```

Output

The sum of 1.5 and 6.3 is 7.8

- Changing this operator, we can subtract **(-)**, multiply **(*)**, divide **(/)**, floor divide **(//)** or find the remainder (%) of two numbers.

Examples: Python Program to Add Two Numbers

```
# Store input numbers
num1 = input('Enter first number: ')
num2 = input('Enter second number: ')

# Add two numbers
sum = float(num1) + float(num2)

# Display the sum
print('The sum of {0} and {1} is {2}'.format(num1, num2, sum))
```

Output

```
Enter first number: 1.5
Enter second number: 6.3
The sum of 1.5 and 6.3 is 7.8
```

Python Input, Output and Import

- Python provides numerous built-in functions that are readily available to us at the Python prompt.
- Some of the functions like input() and print() are widely used for standard input and output operations respectively

Python Output Using print() function

- We use the print() function to output data to the standard output device (screen).

- We can also output data to a file, but this will be discussed later. An example use is given below.

```
print('This sentence is output to the screen’)
# Output: This sentence is output to the screen
a = 5
print('The value of a is', a)
# Output: The value of a is 5
```

Python Input

- To allow input from the user. Python, have the input() function to allow this.
- The syntax for input() is input([prompt])
- where prompt is the string we wish to display on the screen. It is optional.

```
>>> num = input('Enter a number: ')
Enter a number: 10
>>> num
'10'
```

- Here, we can see that the entered value 10 is a string, not a number. To convert this into a number we can use int() or float() functions.

```
>>> int('10')
10
>>> float('10')
10.0
```

Python Input

- This same operation can be performed using the eval() function. But it takes it further. It can evaluate even expressions, provided the input is a string

```
>>> int('2+3')
Traceback (most recent call last):
  File "<string>", line 301, in runcode
  File "<interactive input>", line 1, in <module>
ValueError: invalid literal for int() with base 10: '2+3'
>>> eval('2+3')
5
```

The End
?